

# **UR3 Robotkar, futószalag és ArUco Marker Felismerő Program Dokumentáció**

**Sapientia**

**2023**

**Készítette:**

**Antal József**

# 

**Tartalom**

[UR3 Robotkar, futószalag és ArUco Marker Felismerő Program Dokumentáció 1](#_Toc140441124)

Tartalom [2](#_Toc140441125)

[Bevezető 3](#_Toc140441126)

[Robotkar Program 4](#_Toc140441127)

[1. Mozgatási Pozíciók 6](#_Toc140441128)

[2. Mozgatás és Kommunikáció 6](#_Toc140441129)

[Használat 6](#_Toc140441130)

[Futószalag Program 7](#_Toc140441131)

[Funkciók 8](#_Toc140441132)

[Fő Ciklus 8](#_Toc140441133)

[Használat 8](#_Toc140441134)

[ArUco Marker Felismerő Program 9](#_Toc140441135)

[Használat 10](#_Toc140441136)

# **Bevezető**

A dokumentáció áttekintést nyújt az alkalmazásról, amely egy UR3-as robotkarhoz és egy futószalagon mozgó darabhoz kapcsolódik, alkalmazás lehetővé teszi az ArUco marker felismerését.

A program három fő részből áll: a futószalag vezérléséből, a robotkar vezérléséből és az ArUco marker felismeréséből.

A futószalag program felelős a darabok mozgatásáért a futószalagon. A program időzítőt használva energiaellátást biztosít a motoroknak, és meghatározott idő elteltével leállítja a futószalagot.

A robotkar program vezérli az UR3-as robotkart, és előre meghatározott pozíciókba mozgatja a darabokat. A program a rtde\_control, rtde\_receive és rtde\_io modulokat használja a robotkar vezérléséhez, a pozíciók lekérdezéséhez és a digitális kimenetek beállításához.

Az ArUco Marker Felismerő program lehetővé teszi az ArUco markerek felismerését a darabokon. A program a cv2 és aruco modulokat használja a kamera képének beolvasásához, a markerek felismeréséhez és a kép megjelenítéséhez.

# **Robotkar Program**

A robot.py fájl tartalmazza az UR3-as robotkarhoz kapcsolódó kódot. A program az rtde\_control, rtde\_receive és rtde\_io modulokat használja a robotkar vezérléséhez, a pozíciók lekérdezéséhez és a digitális kimenetek beállításához.

import rtde\_control

import rtde\_receive

import rtde\_io

import time

import subprocess

host = "192.168.98.6"

port = 30004

# control

rtde\_c = rtde\_control.RTDEControlInterface(host)

# getPosition

rtde\_r = rtde\_receive.RTDEReceiveInterface(host)

# gripper

rtde\_io = rtde\_io.RTDEIOInterface(host)

if \_\_name\_\_ == "\_\_main\_\_":

#p[1] += 0.01 #x tengely fix (előre, hátra)

#p[2] -= 0.01 #y tengely fix (fel, le)

#p[3] -= 0.1

#p[4] += 0.1

#p[5] += 0.1

#p[6] += 0.1

 velocity = 1

 acceleration = 1

#TARGET = [Base, Shoulder, Elbow, Wrist1, Wrist2, Wrist3]

target1 = [-0.015, -0.935, 0.85, -1.5, 4.75, 1.6]

target3 = [3, -0.935, 1.2, -1.85, 4.75, 1.6]

rtde\_io.setStandardDigitalOut(0, False)

rtde\_c.moveJ(target1, velocity, acceleration)

subprocess.call(["python", "fp\_elinditas.py"])

p = rtde\_r.getActualTCPPose()

#print(f"pose {p}")

target2 = p

target2[2] -= 0.1

rtde\_c.moveL(target2, velocity, acceleration)

rtde\_io.setStandardDigitalOut(0, True)

rtde\_c.moveJ(target1, velocity, acceleration)

rtde\_c.moveJ(target3, velocity, acceleration)

p = rtde\_r.getActualTCPPose()

#print(f"pose {p}")

target4 = p

target4[2] -= 0.06

rtde\_c.moveL(target4, velocity, acceleration)

rtde\_io.setStandardDigitalOut(0, False)

rtde\_c.stopScript()

**Funkciók**

### **Mozgatási Pozíciók**

A program definiál néhány előre meghatározott pozíciót, amelyekre a robotkar mozgatható. Ezeket a pozíciókat a target1, target2, target3, target4 változókban tároljuk.

### **Mozgatás és Kommunikáció**

A robotkar programban a következő műveletek történnek:

* A robotkar inicializálása és csatlakozás a megadott IP-címhez és portszámhoz.
* Előre meghatározott pozíciókba mozgatás a moveJ és moveL függvények segítségével.
* Digitális kimenetek beállítása a setStandardDigitalOut függvénnyel.

## **Használat**

A robotkar programot a következőképpen lehet futtatni: python robot.py

# **Futószalag Program**

A fp\_elinditas.py fájl tartalmazza a futószalag működését irányító kódot. Ez a program felelős a darabok mozgatásáért a futószalagon.

import subprocess

import time

def ticcmd(\*args):

return subprocess.check\_output(['ticcmd'] + list(args))

# TIC motorvezérlő inicializálása

ticcmd('--exit-safe-start')

# Kezdő időpont

start\_time = time.time()

while True:

# Motor bekapcsolása

ticcmd('--energize')

# Idő ellenőrzése

current\_time = time.time()

elapsed\_time = current\_time - start\_time

# Ellenőrzés, hogy eltelt-e már 15 másodperc

if elapsed\_time >= 17:

break

## **Funkciók**

ticcmd(\*args)

Ez a függvény a ticcmd parancsot futtatja a motorvezérlőn. A paraméterként kapott argumentumokat átadja a ticcmd parancsnak és visszatér a kimenettel.

## **Fő Ciklus**

A futószalag program fő része egy végtelen ciklus, amelyben a motorok be vannak kapcsolva, és az idő ellenőrzése után leállítja a motorokat.

start\_time inicializálása

Ebben a részben az indulási időpontot rögzítjük a program kezdete előtt.

## **Használat**

A futószalag programot a következőképpen lehet futtatni: python fp\_elinditas,py

# **ArUco Marker Felismerő Program**

Az aruco\_marker\_felismero.py fájl tartalmazza az ArUco marker felismerését végző kódot. A program használja a cv2 és aruco modulokat a kamera képének beolvasásához, a markerek felismeréséhez és a kép megjelenítéséhez. Az ArUco marker felismerő program segítségével meghatározható a darabok pontos helye és koordinátái.

import cv2

from cv2 import aruco

import matplotlib.pyplot as plt

cap = cv2.VideoCapture(0)

aruco\_dict = aruco.Dictionary\_get(aruco.DICT\_6X6\_250)

while True:

ret, frame = cap.read()

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

parameters = aruco.DetectorParameters\_create()

corners, ids, rejectedImgPoints = aruco.detectMarkers(gray, aruco\_dict, parameters=parameters)

frame\_markers = aruco.drawDetectedMarkers(frame, corners, ids)

cv2.imshow('ArUco Detection', frame\_markers)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

## **Használat**

A használatához szükséges könyvtárak:

* OpenCV
* rtde

pip install opencv-python rtde

Az ArUco Marker Felismerő programot a következőképpen lehet futtatni:

python aruco\_camera,py